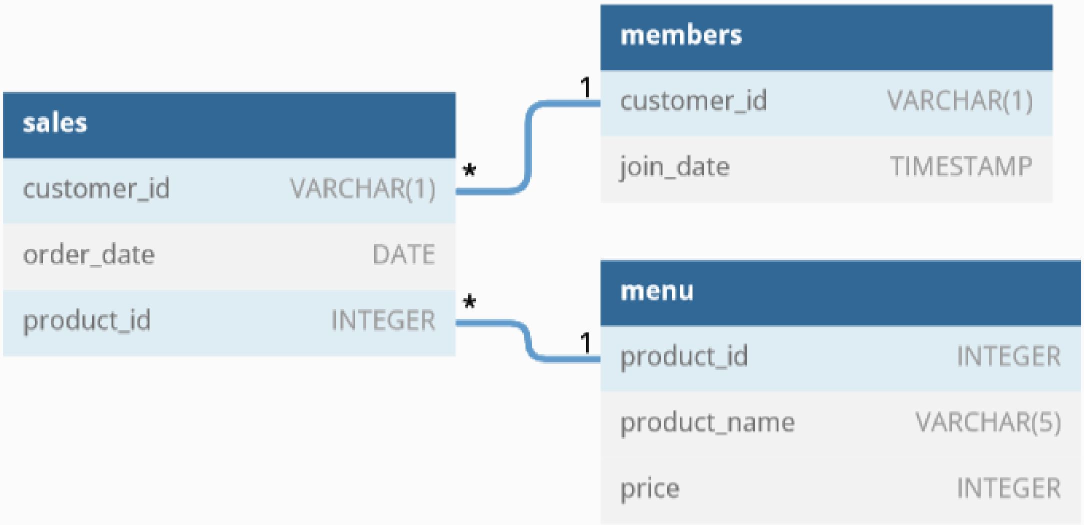
sql

Premise

Three tables - members, menu & sales.



Questions and answers

1. What is the total amount each customer spent at the restaurant?

```SQL
SELECT s.customer_id, SUM(m.price) AS Total_Sales
FROM dbo.sales AS s
JOIN dbo.menu AS m
ON m.product_id = s.product_id
GROUP BY s.customer_id;
```

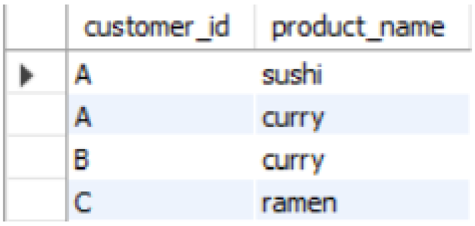
2. How many days has each customer visited the restaurant?

```SQL
SELECT customer_id, COUNT(DISTINCT(order_date)) AS DATE_OF_VISIT
FROM dbo.sales
group by customer_id;
```

3. What was the first item from the menu purchased by each customer?

> We use **partition by, dense_rank() and cte** to solve this.

```sql
WITH most_purchased_cte AS
(
SELECT s.customer_id, m.product_name, s.order_date,
DENSE_RANK() OVER (PARTITION BY s.customer_id
ORDER BY s.order_date) AS First_item
FROM dbo.sales as s
JOIN dbo.menu as m
ON s.product_id = m.product_id
)
SELECT customer_id, product_name
FROM most_purchased_cte
WHERE first_item = 1
GROUP BY customer_id, product_name;
```



customer_id	product_name
A	sushi
A	curry
B	curry
C	ramen

We need to find the first item purchased by each customer.

- ***partition by*** clause will partition table into groups that are having same customer_id.
- ***order by*** will arrange the customers of each partition by "dates".
- ***dense_rank()*** is a window function, which will assign rank in ordered partition of challenges. If two customers have same scores then they will be assigned same rank.

So first, we make a cte which contains the ranking order of items ordered by the customer. Then we only select the first order for each customer. This will give us the solution.

4. What is the most purchased item on the menu and how many times was it purchased by all customers?

> We use **LIMIT** to return only 1 record as the most purchased item

```sql
select m.product_name, count(s.product_id) as most_purchased
from dbo.menu as m
join dbo.sales as s
```

```
on m.product_id = s.product_id
group by s.product_id, m.product_name
order by most_purchased desc
limit 1;
```

5. (Which item was the most popular for each customer?

SQL

```
WITH popular_item_cte AS
(
SELECT s.customer_id, m.product_name,
COUNT(m.product_id) as order_count,
DENSE_RANK() OVER (partition by s.customer_id
ORDER BY COUNT(s.customer_id) DESC) AS popular_item
FROM dbo.sales AS s
JOIN dbo.menu AS m
ON s.product_id = m.product_id
GROUP BY s.customer_id, m.product_name
)
SELECT customer_id, product_name, order_count
FROM popular_item_cte
WHERE popular_item = 1;
```

First try to find the order count and sort the rank in descending order to find most popular item for each customer. Than use the cte to display that.

6. (Which item was purchased first by the customer after they became a member?

order_date should be more than join_date.

SQL

```
with first_item_cte as
(
select s.customer_id, m.join_date, s.order_date, s.product_id,
dense_rank() over (partition by s.customer_id
order by order_date) as ranker
from dbo.sales as s
join dbo.members as m
on s.customer_id = m.customer_id
where s.order_date >= m.join_date
)
select f.customer_id, f.order_date, me.product_name
from first_item_cte as f
join dbo.menu as me
```

```
on f.product_id = me.product_id
where ranker = 1;
```

7. Which item was purchased just before the customer became a member?

similar to Q6, just use desc to sort the last item.

```SQL
with last_item_cte as
(
select s.customer_id, m.join_date, s.order_date, s.product_id,
dense_rank() over (partition by s.customer_id
order by order_date desc) as ranker
from dbo.sales as s
join dbo.members as m
on s.customer_id = m.customer_id
where s.order_date < m.join_date
)
select l.customer_id, l.order_date, me.product_name
from last_item_cte as l
join dbo.menu as me
on l.product_id = me.product_id
where ranker = 1;
```

8. What is the total items and amount spent for each member before they became a member?

```SQL
select s.customer_id, count(s.product_id) as total_items,
sum(me.price) as total_amount
from dbo.sales as s
join dbo.members as m
on s.customer_id = m.customer_id
join dbo.menu as me
on s.product_id = me.product_id
where s.order_date < m.join_date
group by s.customer_id;
```

9. If each $1 spent equates to 10 points and sushi has a 2x points multiplier - how many points would each customer have?

Use Case to select the 2 cases, if it is sushi and the rest, and then use a cte.

SQL

```sql
WITH points_cte AS
(
SELECT * ,
CASE
    WHEN product_id =1 THEN price * 20
    ELSE price * 10
    END AS points
FROM menu
)
SELECT s.customer_id, SUM(p.points) AS Total_points
FROM dbo.sales as s
JOIN points_cte as p
ON s.product_id = p.product_id
GROUP BY s.customer_id;
```

10. In the first week after a customer joins the program (including their join date) they earn 2x points on all items, not just sushi - how many points do customer A and B have at the end of January?

SQL

```sql
WITH dates_cte AS
(
SELECT * ,
DATE_ADD(join_date, INTERVAL 6 DAY) AS Valid_date,
LAST_DAY('2021-01-31') AS  Last_day
FROM dbo.members
)
SELECT d.customer_id, s.order_date, d.join_date,
 d.Valid_date, d.Last_day, m.product_name, m.price,
 SUM(CASE
  WHEN m.product_name = 'sushi' THEN 2 * 10 * m.price
  WHEN s.order_date BETWEEN d.join_date AND d.valid_date THEN 2 * 10 *
m.price
  ELSE 10 * m.price
  END) AS points
FROM dates_cte AS d
JOIN sales AS s
 ON d.customer_id = s.customer_id
JOIN menu AS m
 ON s.product_id = m.product_id
WHERE s.order_date < d.Last_day
```

```
GROUP BY d.customer_id, s.order_date, d.join_date, d.Valid_date,
d.Last_day, m.product_name, m.price
```

Bonus Questions

Join All The Things

```SQL
SELECT s.customer_id, s.order_date, m.product_name, m.price,
CASE
    WHEN me.join_date > s.order_date THEN 'N'
    WHEN me.join_date <= s.order_date THEN 'Y'
    ELSE 'N'
    END AS member
FROM sales as s
JOIN menu as m
ON s.product_id = m.product_id
JOIN members as me
ON s.customer_id = me.customer_id
ORDER BY s.customer_id, s.order_date, m.product_name;
```